Convex Optimization in Python

Y Aditya, G V S S Praneeth Varma and G V V Sharma*

Contents

1 Convex Functions

A single variable function f is said to be convex if

$$f[\lambda x + (1-\lambda)y] \leq \lambda f(x) + (1-\lambda) f(y), \quad (1.1)$$

for $0 < \lambda < 1$.

Problem 1.1. Execute the following python script. Is $\ln x$ convex or concave?

```
import numpy as np
import matplotlib.pyplot as plt


#Plotting log(x)
x = np.linspace(1,8,50)#points on
    the x axis
f=np.log(x)#Objective function
plt.plot(x,f,color=(1,0,1))
plt.grid()
plt.xlabel('$x$')
plt.ylabel('$\ln x$')

#Convexity/Concavity
a = 2
b = 7
lamda = 0.4
c = lamda *a + (1-lamda)*b
f_a = np.log(a)
f_b = np.log(b)

f_c = np.log(c)
f_c_hat = lamda *f_a + (1-lamda)*
    f_b

#Plot commands
plt.plot([a,a],[0,f_a],color
    =(1,0,0),marker='o',label="$f(a)
    $")
plt.plot([b,b],[0,f_b],color
    =(0,1,0),marker='o',label="$f(b)
    $")
plt.plot([c,c],[0,f_c],color
    =(0,0,1),marker='o',label="$f(\
    lambda a + (1-\lambda)b)$")
plt.plot([c,c],[0,f_c_hat],color
    =(1/2,2/3,3/4),marker='o',label=
    "$\lambda f(a) + (1-\lambda)f(b)
    $")
plt.plot([a,b],[f_a,f_b],color
    =(0,1,1))
plt.legend(loc=2)
#plt.savefig('../figs/1.1.eps')
plt.show()#Reveals the plot
```

Problem 1.2. Modify the above python script as follows to plot the parabola $f(x) = x^2$. Is it convex or concave?

```
import numpy as np
import scipy
import matplotlib.pyplot as plt


def sq(x):
```

* The authors are with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

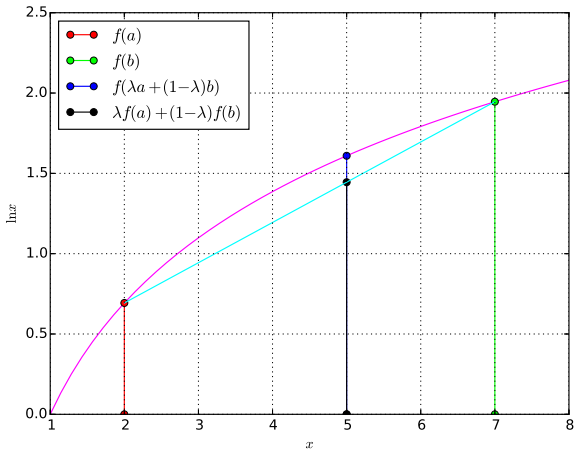

Fig. 1.1: $\ln x$ versus x

.

```
    return x**2

#Plotting the parabola
x = np.linspace(-5,5,50)#points on
    the x axis
vec_sq = scipy.vectorize(sq)
f=vec_sq(x)#Objective function
plt.plot(x,f,color=(1,0,1))
plt.grid()
plt.xlabel('$x$')
plt.ylabel('$\ln x$')

#Convexity/Concavity
a = -3
b = 4
lamda = 0.3
c = lamda *a + (1-lamda)*b
f_a = sq(a)
f_b = sq(b)

f_c = sq(c)
f_c_hat = lamda *f_a + (1-lamda)*
    f_b

#Plot commands
plt.plot([a,a],[0,f_a],color
    =(1,0,0),marker='o',label="$f(a)
    $")
plt.plot([b,b],[0,f_b],color
    =(0,1,0),marker='o',label="$f(b)
    $")
plt.plot([c,c],[0,f_c],color
    =(0,0,1),marker='o',label="$f(\
    lambda a + (1-\lambda)b)$")
plt.plot([c,c],[0,f_c_hat],color
    =(1/2,2/3,3/4),marker='o',label=
    "$\lambda f(a) + (1-\lambda)f(b)
    $")
plt.plot([a,b],[f_a,f_b],color
    =(0,1,1))
plt.legend(loc=2)
#plt.savefig('../figs/1.2.eps')
plt.show()#Reveals the plot
```

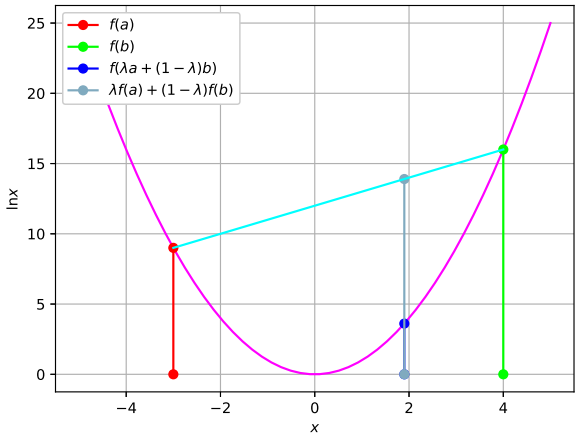

Fig. 1.2: x^2 versus x

.

Problem 1.3. Execute the following script to obtain Fig. 1.3. Comment.

```
import numpy as np
import matplotlib.pyplot as plt


#Plotting log(x)
x = np.linspace(0.5,8,50)#points
    on the x axis
f=np.log(x)#Objective function
plt.plot(x,f,color=(1,0,1))
plt.grid()
plt.xlabel('$x$')
plt.ylabel('$\ln x$')


#Plot commands
#Plotting line segments with x>0
    and y=logx
```

```
#color used to color each line
    with a different color
plt.plot([1,4],[np.log(1),np.log
    (4)],color=(1,0,0),marker='o',
    label="($1$,$\ln(1)$)-($4$,$\ln
    (4)$)")
plt.plot([2,6],[np.log(2),np.log
    (6)],color=(0,1,0),marker='o',
    label="($2$,$\ln(2)$)-($6$,$\ln
    (6)$)")
plt.plot([3,5],[np.log(3),np.log
    (5)],color=(0,0,1),marker='o',
    label="($3$,$\ln(3)$)-($5$,$\ln
    (5)$)")
plt.legend(loc=2)
#plt.savefig('../figs/1.3.eps')
plt.show()#Reveals the plot
```

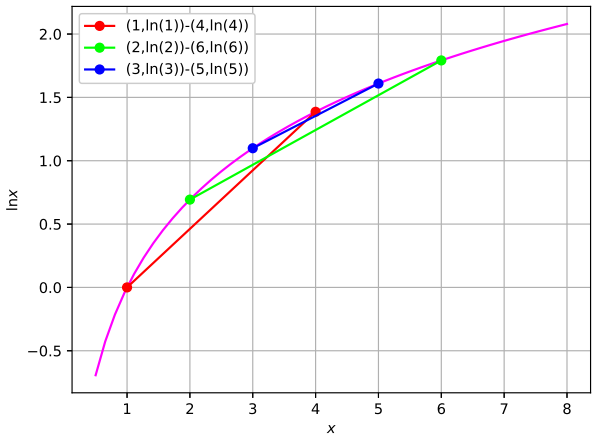

Fig. 1.3: Segments are below the curve

.

Problem 1.4. Modify the script in the previous problem for $f(x) = x^2$. What can you conclude?

Problem 1.5. Let

$$f(\mathbf{x}) = x_1 x_2, \quad \mathbf{x} \in \mathbf{R}^2 \tag{1.2}$$

Sketch $f(\mathbf{x})$ and deduce whether it is convex. Can you theoretically explain your observation using (1.1)?

2 Convex Optimization

2.1 Karush Kuhn-Tucker Conditions

Problem 2.1. Plot the circles

$$f(\mathbf{x}) = (x_1 - 8)^2 + (x_2 - 6)^2 = r^2 \tag{2.1}$$

$\mathbf{x} = (x_1, x_2)^T$, for different values of r along with the line

$$g(\mathbf{x}) = x_1 + x_2 - 9 = 0 \tag{2.2}$$

using the following program. From the graph, find

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{s.t.} \tag{2.3}$$

$$g(\mathbf{x}) = x_1 + x_2 - 9 = 0 \tag{2.4}$$

```
import numpy as np
import matplotlib.pyplot as plt

#Plotting the circle
x = 8*np.ones(8)
y = 6*np.ones(8)
r = np.arange(8)/np.sqrt(2)
phi = np.linspace(0.0,2*np.pi,100)
na=np.newaxis
# the first axis of these arrays
    varies the angle,
# the second varies the circles
x_line = x[na,:]+r[na,:]*np.sin(
    phi[:,na])
y_line = y[na,:]+r[na,:]*np.cos(
    phi[:,na])
ax=plt.plot(x_line,y_line,'-')

#Plotting the line
x1 = np.linspace(0,10,100)
x2 = 9*np.ones(100) - x1
bx=plt.plot(x1,x2,label="$x_1+x_2
    -9=0$")
plt.axis('equal')
plt.grid()
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.legend([ax[5], bx[0]],['$(x_1
    -8)^2+(x_2-6)^2=\\frac{25}{2}$'
    ,'$x_1+x_2-9=0$'], loc='best')

#plt.savefig('../figs/2.1.eps')
plt.show()
```

Problem 2.2. Obtain a theoretical solution for problem 2.1 using coordinate geometry.

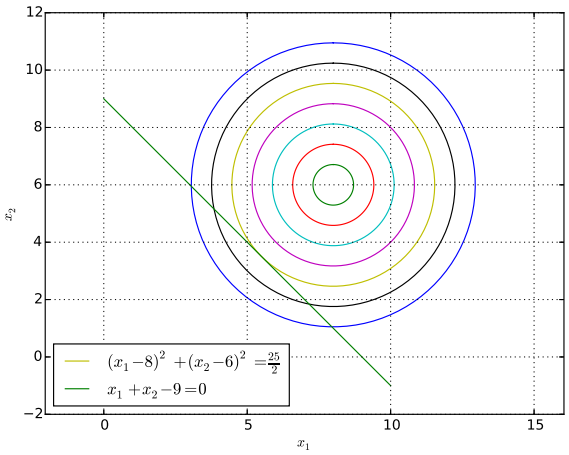

Fig. 2.1: Finding $\min_{\mathbf{x}} f(\mathbf{x})$

.

Solution: From (2.2) and (2.1),

$$r^2 = (x_1 - 8)^2 + (3 - x_1)^2 \tag{2.5}$$

$$= 2x_1^2 - 22x_1 + 73 \tag{2.6}$$

$$\Rightarrow r^2 = \frac{(2x_1 - 11)^2 + 5^2}{2} \tag{2.7}$$

which is minium when $x_1 = \frac{11}{2}, x_2 = \frac{7}{2}$. The minimum value is $\frac{25}{2}$ and the radius $r = \frac{5}{\sqrt{2}}$.

Problem 2.3. Define

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda g(\mathbf{x}) \tag{2.8}$$

and

$$\nabla = \begin{pmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial \lambda} \end{pmatrix} \tag{2.9}$$

Solve the equations

$$\nabla L(\mathbf{x}, \lambda) = 0. \tag{2.10}$$

How is this related to problem 2.1? What is the sign of λ? L is known as the Lagrangian and the above technique is known as the Method of Lagrange Multipliers.

Solution: From (2.2) and (2.1),

$$L(\mathbf{x}, \lambda) = (x_1 - 8)^2 + (x_2 - 6)^2 - \lambda(x_1 + x_2 - 9) \tag{2.11}$$

$$\Rightarrow \nabla L(\mathbf{x}, \lambda) = \begin{pmatrix} 2x_1 - 16 - \lambda \\ 2x_2 - 12 - \lambda \\ x_1 + x_2 - 9 \end{pmatrix} \tag{2.12}$$

$$= \begin{pmatrix} 2 & 0 & -1 \\ 0 & 2 & -1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \lambda \end{pmatrix} = \begin{pmatrix} 16 \\ 12 \\ 9 \end{pmatrix} = 0 \tag{2.13}$$

$$\Rightarrow \begin{pmatrix} x_1 \\ x_2 \\ \lambda \end{pmatrix} = \begin{pmatrix} \frac{11}{2} \\ \frac{7}{2} \\ -5 \end{pmatrix} \tag{2.14}$$

using the following python script. Note that this method yields the same result as the previous exercises. Thus, λ is negative.

```
import numpy as np

A = np.matrix('2 0 -1; 0 2 -1; 1 1
    0')
b = np.matrix('16 ; 12 ; 9')
print (np.linalg.inv(A)*b)
```

Problem 2.4. Modify the code in problem 2.1 to find a graphical solution for minimising

$$f(\mathbf{x}) = (x_1 - 8)^2 + (x_2 - 6)^2 \tag{2.15}$$

with constraint

$$g(\mathbf{x}) = x_1 + x_2 - 9 \geq 0 \tag{2.16}$$

Solution: This problem reduces to finding the radius of the smallest circle in the shaded area in Fig. 2.4 . It is clear that this radius is 0.

```
import numpy as np
import matplotlib.pyplot as plt


#Plotting the circle
x = 8*np.ones(8)
y = 6*np.ones(8)
r = np.arange(8)/np.sqrt(2)
phi = np.linspace(0.0,2*np.pi,100)
na=np.newaxis
# the first axis of these arrays
    varies the angle,
# the second varies the circles
```

```
x_line = x[na,:]+r[na,:]*np.sin(
    phi[:,na])
y_line = y[na,:]+r[na,:]*np.cos(
    phi[:,na])

ax=plt.plot(x_line,y_line,'-')

#Plotting the line
x1 = np.linspace(-5,10,100)
x2 = 9*np.ones(100) - x1

x1_extend = np.linspace(0,20,100)
x2_extend = 22*np.ones(100) -
    x1_extend

bx=plt.plot(x1,x2,label="$x_1+x_2
    -9=0$")

plt.fill_between(x1_extend,
    x2_extend,color='grey')
plt.fill_between(x1,x2,color='
    white')
plt.axis('equal')
plt.grid()
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.legend( loc='best')

plt.xlim(-5, 15)
plt.ylim(0, 22)


#plt.savefig('../figs/2.4.eps')
plt.show()
```

Problem 2.5. Now use the method of Lagrange multipliers to solve problem 2.4 and compare with the graphical solution. Comment.

Solution: Using the method of Lagrange multipliers, the solution is the same as the one obtained in problem 2.4, which is different from the graphical solution. This means that the Lagrange multipliers method cannot be applied blindly.

Problem 2.6. Repeat problem 2.5 by keeping $\lambda = 0$. Comment.

Solution: Keeping $\lambda = 0$ results in $x_1 = 8, x_2 = 6$, which is the correct solution. The minimum value of $f(\mathbf{x})$ without any constraints lies in the region

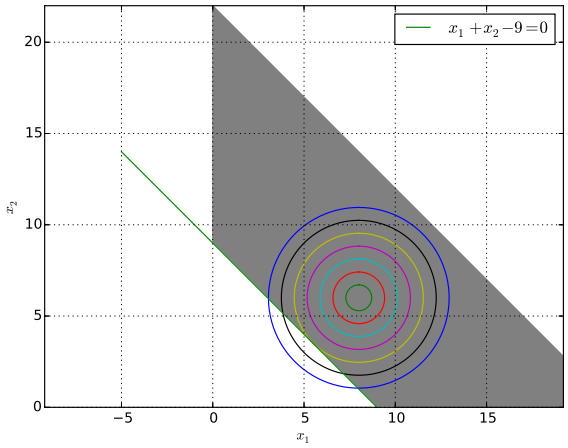

Fig. 2.4: Smallest circle in the shaded region is a point.

$g(\mathbf{x}) = 0$. In this case, $\lambda = 0$.

Problem 2.7. Find a graphical solution for minimising

$$f(\mathbf{x}) = (x_1 - 8)^2 + (x_2 - 6)^2 \tag{2.17}$$

with constraint

$$g(\mathbf{x}) = x_1 + x_2 - 9 \leq 0. \tag{2.18}$$

Summarize your observations.

Solution: In Fig. 2.7, the shaded region represents the constraint. Thus, the solution is the same as the one in problem 2.4. This implies that the method of Lagrange multipliers can be used to solve the optimization problem with this inequality constraint as well. Table 2.7 summarizes the conditions for this based on the observations so far.

```
import numpy as np
import matplotlib.pyplot as plt


#Plotting the circle
x = 8*np.ones(8)
y = 6*np.ones(8)
r = np.arange(8)/np.sqrt(2)
phi = np.linspace(0.0,2*np.pi,100)
na=np.newaxis
# the first axis of these arrays
    varies the angle,
# the second varies the circles
x_line = x[na,:]+r[na,:]*np.sin(
```

```
    phi[:,na])
y_line = y[na,:]+r[na,:]*np.cos(
    phi[:,na])

ax=plt.plot(x_line,y_line,'-')

#Plotting the line
x1 = np.linspace(0,10,100)
x2 = 9*np.ones(100) - x1

bx=plt.plot(x1,x2,label="$x_1+x_2
    -9=0$")
d =np.zeros(len(x2))
plt.fill_between(x1,x2,where=x2>=d
    ,color='grey')
plt.axis('equal')
plt.grid()
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.legend( loc='best')

#plt.savefig('../figs/2.7.eps')
plt.show()
```

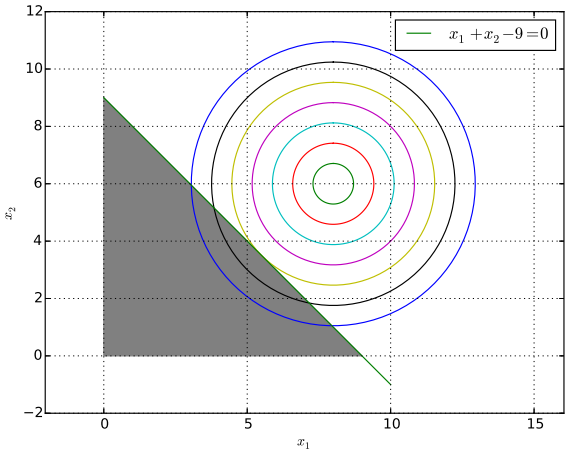

Fig. 2.7: Finding $\min_{\mathbf{x}} f(\mathbf{x})$.

TABLE 2.7: Summary of conditions.

Cost	Constraint	λ
$f(\mathbf{x})$	$g(\mathbf{x}) = 0$	< 0
	$g(\mathbf{x}) \geq 0$	0
	$g(\mathbf{x}) \leq 0$	< 0

Problem 2.8. Find a graphical solution for

$$\min_{\mathbf{x}} f(\mathbf{x}) = (x_1 - 8)^2 + (x_2 - 6)^2 \tag{2.19}$$

with constraint

$$g(\mathbf{x}) = x_1 + x_2 - 18 = 0 \tag{2.20}$$

Solution:

```
import numpy as np
import matplotlib.pyplot as plt

#Plotting the circle
x = 8*np.ones(8)
y = 6*np.ones(8)
r = np.arange(8)/np.sqrt(2)
phi = np.linspace(0.0,2*np.pi,100)
na=np.newaxis
# the first axis of these arrays
    varies the angle,
# the second varies the circles
x_line = x[na,:]+r[na,:]*np.sin(
    phi[:,na])
y_line = y[na,:]+r[na,:]*np.cos(
    phi[:,na])
ax=plt.plot(x_line,y_line,'-')

#Plotting the line
x1 = np.linspace(5,15,100)
x2 = 18*np.ones(100) - x1
bx=plt.plot(x1,x2)
plt.axis('equal')
plt.grid()
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.legend([ax[4], bx[0]],['$(x_1
    -8)^2+(x_2-6)^2=8$','$x_1+x_2
    -18=0$'], loc='best')

#plt.savefig('../figs/2.8.eps')
plt.show()
```

Problem 2.9. Repeat problem 2.8 using the method of Lagrange mutipliers. What is the sign of λ?

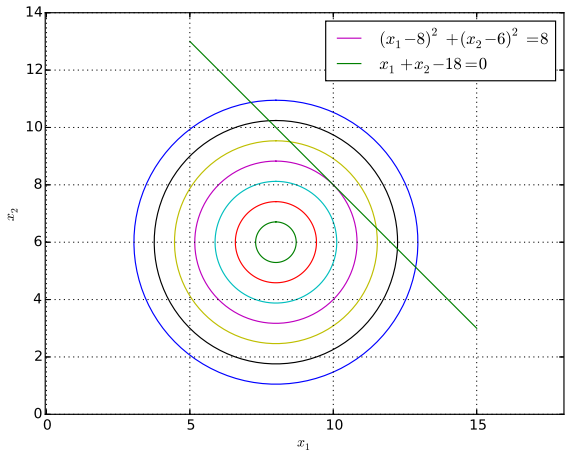

Fig. 2.8: Finding $\min_{\mathbf{x}} f(\mathbf{x})$.

Solution: From (2.25) and (2.25),

$$L(\mathbf{x},\lambda) = (x_1-8)^2 + (x_2-6)^2 - \lambda(x_1+x_2-18) \tag{2.21}$$

$$\Rightarrow \nabla L(\mathbf{x},\lambda) = \begin{pmatrix} 2x_1 - 16 - \lambda \\ 2x_2 - 12 - \lambda \\ x_1 + x_2 - 18 \end{pmatrix} \tag{2.22}$$

$$= \begin{pmatrix} 2 & 0 & -1 \\ 0 & 2 & -1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \lambda \end{pmatrix} = \begin{pmatrix} 16 \\ 12 \\ 18 \end{pmatrix} = 0 \tag{2.23}$$

$$\Rightarrow \begin{pmatrix} x_1 \\ x_2 \\ \lambda \end{pmatrix} = \begin{pmatrix} 10 \\ 8 \\ 4 \end{pmatrix} \tag{2.24}$$

using the following python script. Thus, λ is positive and the minimum value of f is 8.

```
import numpy as np

A = np.matrix('2 0 -1; 0 2 -1; 1 1
    0')
b = np.matrix('16 ; 12 ; 18')
print (np.linalg.inv(A)*b)
```

Problem 2.10. Solve

$$\min_{\mathbf{x}} f(\mathbf{x}) = (x_1-8)^2 + (x_2-6)^2 \tag{2.25}$$

with constraint

$$g(\mathbf{x}) = x_1 + x_2 - 18 \geq 0 \tag{2.26}$$

Solution: Since the unconstrained solution is outside the region $g(\mathbf{x}) \geq 0$, the solution is the same as the one in problem 2.8.

Problem 2.11. Based on the problems so far, generalise the Lagrange multipliers method for

$$\min_{\mathbf{x}} f(\mathbf{x}), \quad g(\mathbf{x}) \geq 0 \tag{2.27}$$

Solution: Considering $L(\mathbf{x},\lambda) = f(\mathbf{x}) - \lambda g(\mathbf{x})$, for $g(\mathbf{x}) = x_1 + x_2 - 18 \geq 0$ we found $\lambda > 0$ and for $g(\mathbf{x}) = x_1 + x_2 - 9 \leq 0, \lambda < 0$. A single condition can be obtained by framing the optimization problem as

$$\min_{\mathbf{x}} f(\mathbf{x}), \quad g(\mathbf{x}) \leq 0 \tag{2.28}$$

with the Lagrangian

$$L(\mathbf{x},\lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x}), \tag{2.29}$$

provided

$$\nabla L(\mathbf{x},\lambda) = 0 \Rightarrow \lambda > 0 \tag{2.30}$$

else, $\lambda = 0$.

Problem 2.12. Solve

$$\min_{\mathbf{x}} f(\mathbf{x}) = 4x_1^2 + 2x_2^2 \tag{2.31}$$

with constraints

$$g_1(\mathbf{x}) = 3x_1 + x_2 - 8 = 0 \tag{2.32}$$

$$g_2(\mathbf{x}) = 15 - 2x_1 - 4x_2 \geq 0 \tag{2.33}$$

Solution: Considering the Lagrangian

$$L(\mathbf{x},\lambda) = f(\mathbf{x}) + \lambda g_1(\mathbf{x}) - \mu g_2(\mathbf{x}) \tag{2.34}$$

$$= 4x_1^2 + 2x_2^2 + \lambda(3x_1 + x_2 - 8) - \mu(15 - 2x_1 - 4x_2), \tag{2.35}$$

$$\nabla L(\mathbf{x},\lambda) = \begin{pmatrix} 8x_1 + 3\lambda + 2\mu \\ 4x_2 + \lambda + 4\mu \\ 3x_1 + x_2 - 8 \\ -2x_1 - 4x_2 + 15 \end{pmatrix} = 0 \tag{2.36}$$

resulting in the matrix equation

$$\implies \begin{pmatrix} 8 & 0 & 3 & 2 \\ 0 & 4 & 1 & 4 \\ 3 & 1 & 0 & 0 \\ 2 & 4 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \lambda \\ \mu \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 8 \\ 15 \end{pmatrix} \tag{2.37}$$

$$\implies \begin{pmatrix} x_1 \\ x_2 \\ \lambda \\ \mu \end{pmatrix} = \begin{pmatrix} 1.7 \\ 2.9 \\ -3.12 \\ -2.12 \end{pmatrix} \tag{2.38}$$

using the following python script. The (incorrect) graphical solution is available in Fig. 2.12

```
import numpy as np
import matplotlib.pyplot as plt


#Theoretical solution using
    Lagrange multipliers
A = np.matrix('8 0 3 2; 0 4 1 4; 3
    1 0 0; 2 4 0 0')
b = np.matrix('0 ; 0 ; 8; 15')
sol = np.array(np.linalg.inv(A)*b)
r = np.sqrt(4*sol[0]**2+2*sol
    [1]**2)

print (sol)
print (r)

#Plotting the ellipse f
phi = np.linspace(0.0,2*np.pi,100)

alpha = r/2
beta  = r/np.sqrt(2)
x_line = alpha*np.cos(phi)
y_line = beta*np.sin(phi)

plt.plot(x_line,y_line,'b',label='
    $f(\mathbf{x})=4x_1^2 + 2x_2^2 =
    r^2$')

#Plotting g1
x1 = np.linspace(-5,5,100)
x2 = 8*np.ones(100)  - 3*x1
plt.plot(x1,x2,'g',label='$g_1(\
    mathbf{x})=3x_1 + x_2 = 8$')

#Plotting g2
y1 = np.linspace(-5,5,100)
y2 = (15*np.ones(100) - 2*y1)/4
plt.plot(y1,y2,'r',label='$g_2(\
    mathbf{x})=4x_1 + 2x_2 = 15$')

plt.axis('equal')
plt.grid()
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.legend(loc=3,prop={'size':12})

plt.xlim(-5, 5)
plt.ylim(-5, 5)

#Display solution
A = sol[0]
B = sol[1]

plt.plot(A,B,'o')
for i,j in zip(A,B):
    plt.annotate('%s)' %j, xy=(i,j
        ), xytext=(30,0), textcoords
        ='offset points')
    plt.annotate('(%s,' %i, xy=(i,
        j))

#plt.savefig('../figs/2.12.eps')
plt.show()
```

Note that $\mu < 0$, contradicting the necessary condition in (2.30).

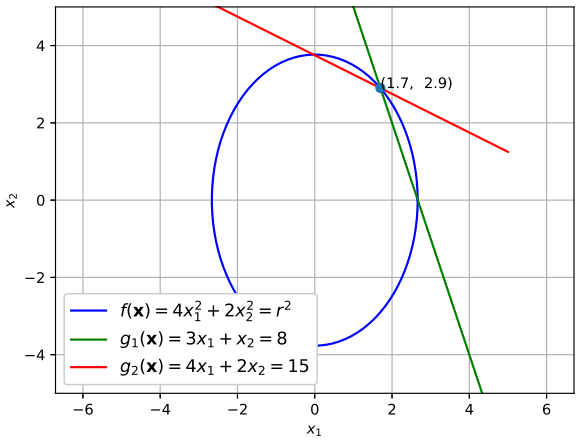

Fig. 2.12: Incorrect solution is at intersection of all curves $r = 5.33$

Problem 2.13. Obtain the correct solution to the previous problem by considering $\mu = 0$.

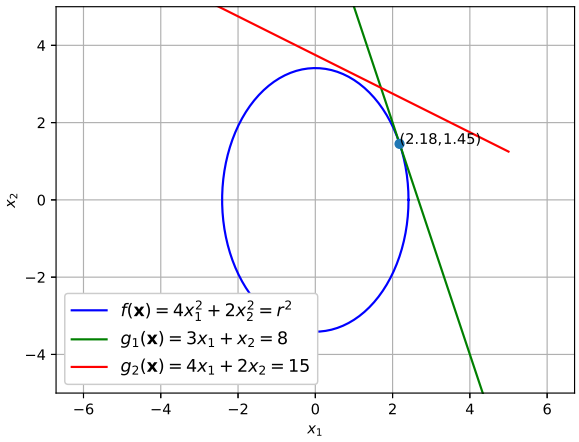

Fig. 2.13: Optimal solution is where $g_1(x)$ touches the curve $r = 4.82$

Problem 2.14. Solve

$$\min_{\mathbf{x}} f(\mathbf{x}) = 4x_1^2 + 2x_2^2 \tag{2.39}$$

with constraints

$$g_1(\mathbf{x}) = 3x_1 + x_2 - 8 = 0 \tag{2.40}$$
$$g_2(\mathbf{x}) = 15 - 2x_1 - 4x_2 \leq 0 \tag{2.41}$$

Problem 2.15. Based on whatever you have done so far, list the steps that you would use in general for solving a convex optimization problem like (2.39) using Lagrange Multipliers. These are called Karush-Kuhn-Tucker(KKT) conditions.

Solution: For a problem defined by

$$\mathbf{x}^* = \min_{\mathbf{x}} f(\mathbf{x}) \tag{2.42}$$
$$\text{subject to } h_i(\mathbf{x}) = 0, \forall i = 1, .., m \tag{2.43}$$
$$\text{subject to } g_i(\mathbf{x}) \leq 0, \forall i = 1, .., n \tag{2.44}$$

the optimal solution is obtained through

$$\mathbf{x}^* = \min_{\mathbf{x}} L(\mathbf{x}, \lambda, \mu) \tag{2.45}$$
$$= \min_{\mathbf{x}} f(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i h_i(\mathbf{x}) + \sum_{i=1}^{n} \mu_i g_i(\mathbf{x}), \tag{2.46}$$

using the KKT conditions

$$\Rightarrow \nabla_{\mathbf{x}} f(\mathbf{x}) + \sum_{i=1}^{m} \nabla_{\mathbf{x}} \lambda_i h_i(\mathbf{x}) + \sum_{i=1}^{n} \mu_i \nabla_{\mathbf{x}} g_i(\mathbf{x}) = 0 \tag{2.47}$$
$$\text{subject to } \mu_i g_i(\mathbf{x}) = 0, \forall i = 1, .., n \tag{2.48}$$
$$\text{and } \mu_i \geq 0, \forall i = 1, .., n \tag{2.49}$$

Problem 2.16. Maxmimize

$$f(\mathbf{x}) = \sqrt{x_1 x_2} \tag{2.50}$$

with the constraints

$$x_1^2 + x_2^2 \leq 5 \tag{2.51}$$
$$x_1 \geq 0, x_2 \geq 0 \tag{2.52}$$

Problem 2.17. Solve

$$\min_{\mathbf{x}} \quad x_1 + x_2 \tag{2.53}$$

with the constraints

$$x_1^2 - x_1 + x_2^2 \leq 0 \tag{2.54}$$

where $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

Solution: Using the method of Lagrange multipliers,

$$\nabla \{f(\mathbf{x}) + \mu g(\mathbf{x})\} = 0, \quad \mu \geq 0 \tag{2.55}$$

resulting in the equations

$$2x_1\mu - \mu + 1 = 0 \tag{2.56}$$
$$2x_2\mu + 1 = 0 \tag{2.57}$$
$$x_1^2 - x_1 + x_2^2 = 0 \tag{2.58}$$

which can be simplified to obtain

$$\left(\frac{1-\mu}{2\mu}\right)^2 + \left(\frac{1}{2\mu}\right)^2 + \frac{1-\mu}{2\mu} = 0 \tag{2.59}$$
$$\Rightarrow 1 + \mu^2 - 2\mu + 1 + 2\mu(1-\mu) = 0 \tag{2.60}$$
$$\Rightarrow \mu^2 = 2, or \mu = \pm\sqrt{2} \tag{2.61}$$

From (2.39), $\mu \geq 0 \Rightarrow \mu = \sqrt{2}$. The desired solution is

$$\mathbf{x} = \begin{pmatrix} \frac{\sqrt{2}-1}{2\sqrt{2}} \\ -\frac{1}{2\sqrt{2}} \end{pmatrix} \tag{2.62}$$

Graphical solution: The constraint can be expressed

as

$$x_1^2 - x_1 + x_2^2 \leq 0 \tag{2.63}$$

$$\Rightarrow \left(x_1 - \frac{1}{2}\right)^2 + x_2^2 \leq \left(\frac{1}{2}\right)^2 \tag{2.64}$$

```
import numpy as np
import matplotlib.pyplot as plt


sol = np.zeros((2,1))
#Printing minimum
sol[0] = (np.sqrt(2)-1)/(2*np.sqrt
    (2))
sol[1] = -1/(2*np.sqrt(2))


#Plotting the circle

circle = plt.Circle((0.5, 0), 0.5,
     color='r', fill=False)

fig, ax = plt.subplots() # note we
     must use plt.subplots, not plt.
    subplot

ax.add_artist(circle)

#Display solution
A = np.around(sol[0],decimals=2)
B = np.around(sol[1],decimals=2)


plt.plot(A,B,'o')
for xy in zip(A,B):
        ax.annotate('(%s, %s)' %
           xy, xy=xy, textcoords='
           data')

print (sol)


#Plotting the line
p = (sol[0]+sol[1])*np.arange
    (-2,3)
x = np.linspace(0,1,100)
na=np.newaxis

x_line = x[:,na]
y_line = p[na,:] - x[:,na]
bx=plt.plot(x_line,y_line,'-')


plt.axis('equal')
plt.grid()
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.ylim(-0.5,0.5)

plt.legend([bx[3]],['$x_1+x_2=\\
    frac{\sqrt{2}-2}{2\sqrt{2}}$'],
    loc='best',prop={'size':11})
#plt.savefig('../figs/2.15.eps')
plt.show()
```

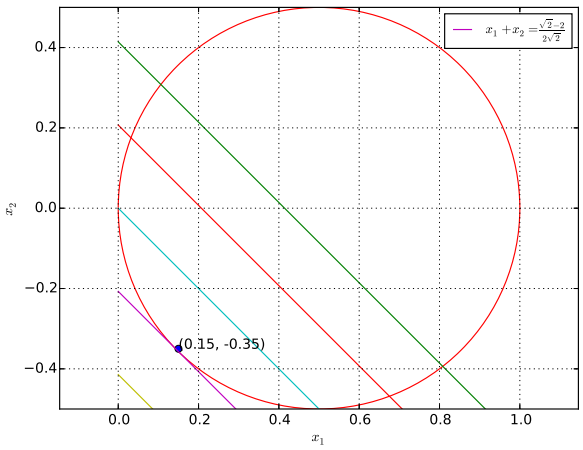

Fig. 2.17: Optimal solution is the lower tangent to the circle

3 Semi-definite Programming

Problem 3.1. The problem

$$\min_{\mathbf{x}} x_{11} + x_{12} \tag{3.1}$$

with constraints

$$x_{11} + x_{22} = 1 \tag{3.2}$$

$$\begin{pmatrix} x_{11} & x_{12} \\ x_{12} & x_{22} \end{pmatrix} \geq 0 \quad (\geq \text{ means positive definite}) \tag{3.3}$$

is known as a semi-definite program. Find a numerical solution to this problem. Compare with the solution in problem 2.17.

Solution: The *cvxopt* solver needs to be used in order to find a numerical solution. For this, the given problem has to be reformulated as

$$\min_{\mathbf{x}} \begin{pmatrix} 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_{11} \\ x_{12} \\ x_{22} \end{pmatrix} \quad \text{s.t} \tag{3.4}$$

$$\begin{pmatrix} 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{11} \\ x_{12} \\ x_{22} \end{pmatrix} = 1 \tag{3.5}$$

$$x_{11} \begin{pmatrix} -1 & 0 \\ 0 & 0 \end{pmatrix} + x_{12} \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} + x_{22} \begin{pmatrix} 0 & 0 \\ 0 & -1 \end{pmatrix} \leq \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}. \tag{3.6}$$

The following script provides the solution to this problem.

```
from cvxopt import matrix
from cvxopt import solvers

c = matrix([1.,1.,0.])
G = [ matrix([[-1., 0., 0., 0.],[
    0., -1., -1., 0.],[0., 0., 0.,
     -1.]]) ]

Aval = matrix([1.,0.,1.],(1,3))
bval = matrix([1.])

h = [ matrix([[0., 0.], [0., 0.]])
    ]
sol = solvers.sdp(c, Gs=G, hs=h,A=
    Aval, b=bval)
print(sol['x'])
```

Problem 3.2. Show that problem 3.1 is equivalent to problem 2.17.

Problem 3.3. Minimize

$$-x_{11} - 2x_{12} - 5x_{22} \tag{3.7}$$

subject to

$$2x_{11} + 3x_{12} + x_{22} = 7 \tag{3.8}$$

$$x_{11} + x_{12} \geq 1 \tag{3.9}$$

$$x_{11}, x_{12}, x_{22} \geq 0 \tag{3.10}$$

$$\begin{pmatrix} x_{11} & x_{12} \\ x_{12} & x_{22} \end{pmatrix} \geq 0 \tag{3.11}$$

using *cvxopt*.

Problem 3.4. Repeat the above exercise by converting the problem into a convex optimization problem in two variables and using graphical plots.

Problem 3.5. Solve the above problem using the KKT conditions. Comment.

4 Linear Programming

Problem 4.1. Graphically obtain a solution to the following

$$\max_{\mathbf{x}} 6x_1 + 5x_2 \tag{4.1}$$

with constraints

$$x_1 + x_2 \leq 5 \tag{4.2}$$

$$3x_1 + 2x_2 \leq 12 \tag{4.3}$$

$$\text{where } x_1, x_2 \geq 0 \tag{4.4}$$

Solution: The following program plots the solution in Fig. 4.1

```
import numpy as np
import mpmath as mp
import matplotlib.pyplot as plt
from scipy.spatial import
    ConvexHull




#Vertices of the convex polygon
    formed by intersection of the
    constraints
points = np.array([[0, 0], [0,
    5],[2, 3], [4,0]])
#Filling up the polygon
plt.fill(points[:,0], points[:,1],
     'k', alpha=0.3)



#Plotting the set of possible cost
     functions

p = 9*np.arange(1,5)
print (p)
x = np.linspace(0,4,100);
na=np.newaxis

x_line = x[:,na]
y_line = p[na,:]/5.0 - 6.0/5.0*x
    [:,na]
```

```
bx=plt.plot(x_line,y_line,'-')



#Plotting the first constraint
x1 = np.linspace(0,4,100)
x2 = 5*np.ones(100) - x1

ax=plt.plot(x1,x2)


#plotting the second constraint
x2 = 6*np.ones(100) - 3.0/2.0*x1

cx=plt.plot(x1,x2)

#Displaying solution on graph
sol = np.zeros((2,1))
sol[0] = 2
sol[1] = 3

#Display solution
A = sol[0]
B = sol[1]


plt.plot(A,B,'o')
for xy in zip(A,B):
        plt.annotate('(%s, %s)' %
           xy, xy=xy, textcoords='
           data')

plt.grid()
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.legend([bx[2], ax[0], cx[0]],[
   '$6x_1+5x_2=27$', '$x_1+x_2=5$',
   '$3x_1+2x_2=12$'], loc='best',
   prop={'size':11})
plt.xlim(0,4)
plt.ylim(0,5)
plt.savefig('../figs/4.1.eps')
plt.show()
```

Problem 4.2. Now use *cvxopt* to obtain a solution to problem 4.1.

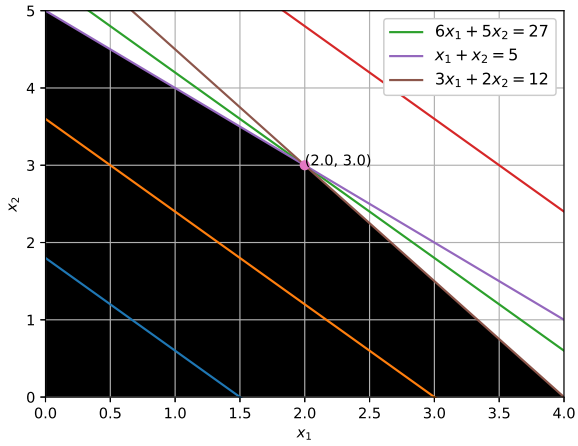

Fig. 4.1: The cost function intersects with the two constraints at $\mathbf{x} = (2, 3)$.

Solution: The given problem is expressed as follows

$$\min_{\mathbf{x}} \mathbf{c}^T\mathbf{x} \quad s.t. \tag{4.5}$$

$$\mathbf{Ax} \preceq \mathbf{b} \tag{4.6}$$

where

$$\mathbf{c} = \begin{pmatrix} -6 \\ -5 \end{pmatrix}, \mathbf{A} = \begin{pmatrix} 1 & 1 \\ 3 & 2 \\ -1 & 0 \\ 0 & -1 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} 5 \\ 12 \\ 0 \\ 0 \end{pmatrix} \tag{4.7}$$

The desired solution is then obtained using the following program.

```
from cvxopt import matrix
from cvxopt import solvers

A = matrix([ [1.0, 3.0, -1.0, 0],
   [1.0, 2.0, 0, -1.0] ])
b = matrix([ 5.0, 12.0, 0.0, 0.0
   ])
c = matrix([ -6.0, -5.0 ])

sol = solvers.sdp(c, A, b)
print(sol['x'])
```

Problem 4.3. Verify your solution to the above problem using the method of Lagrange multipliers.

Problem 4.4. Maximise $5x_1 + 3x_2$ w.r.t the con-

straints

$$
\begin{aligned}
x_1 + x_2 &\leq 2 \\
5x_1 + 2x_2 &\leq 10 \\
3x_1 + 8x_2 &\leq 12 \\
\text{where } x_1, x_2 &\geq 0
\end{aligned}
$$